12

DNA 4266F

IMPLEMENTATION STUDY OF SAM-IV ON THE ILLIAC-IV COMPUTER

Mathematical Applications Group, Inc.
3 Westchester Plaza
Elmsford, New York 10523

February 1977

Final Report

CONTRACT No. DNA 001-75-C-0025

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

THIS WORK SPONSORED BY THE DEFENSE ADVANCED RESEARCH PROJECTS AGENCY UNDER DARPA ORDER No. 1962-4.

Prepared for
Director
DEFENSE NUCLEAR AGENCY
Washington, D. C. 20305

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER DNA 4266F ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 IMPLEMENTATION STUDY OF SAM-~~IV~~ ON THE ILLIAC-~~IV~~ COMPUTER.	9	5. TYPE OF REPORT & PERIOD COVERED Final ~~Report~~,
	14 MAGI-	6. PERFORMING ORG. REPORT NUMBER M-7053
7. AUTHOR(s) 10 Eugene S./Troubetzkoy, Malvin H./Kalos Herbert A./Steinberg	15	8. CONTRACT OR GRANT NUMBER(s) DNA 001-75-C-0025
9. PERFORMING ORGANIZATION NAME AND ADDRESS Mathematical Applications Group, Inc. ✓ 3 Westchester Plaza Elmsford, New York 10523		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DARPA Order ~~No.~~ 1962-4
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209	11	12. REPORT DATE Feb~~ruary~~ 1977 13. NUMBER OF PAGES 20
14. MONITORING AGENCY NAME & ADDRESS (if different ~~from~~ Controlling Office) Director Defense Nuclear Agency Washington, D.C. 20305	12 17p.	15. SECURITY CLASS (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

18 DNA, SBIE

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

19 4266

AD-E300 019

18. SUPPLEMENTARY NOTES

This work sponsored by the Defense Advanced Research Projects Agency under DARPA Order No. 1962-4.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Monte Carlo
ILLIAC-IV
Radiation Penetration
Parallel Computer

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A promising technique to parallelize Monte Carlo calculations has been developed previously and implemented in a versatile program to study radiation penetration through complex geometry. Attempts to make the program operational on the ILLIAC-IV computer were successful in a very limited sense, as only a few short runs ran to completion. The tentative conclusions drawn from this limited experience are not in contradiction with the validity of the technique parallelization.

TABLE OF CONTENTS

I. INTRODUCTION

Under Contract DNA001-72-C-0057, MAGI has completed the design and implementation of a preliminary version of a Monte Carlo three-dimensional, time-dependent radiation penetration in complex geometry code for the ILLIAC IV. The design should serve as a guide to the programming of complex Monte Carlo codes for any parallel computers, and, with some modification, for vector computers. The implementation was supposed to show the feasibility of such a design and the efficiency of the design. It was also supposed to produce a valuable tool for radiation penetration studies.

Under the present contract, work has been done towards the inclusion of the treatment of neutrons and towards other improvements of the code. The major effort, however, turned out in attempts to make the preliminary version more or less operational on the actual ARRAY processor of the ILLIAC IV, rather than under translation or under simulation on conventional computers. From the few short runs which we were able to complete, we extrapolate, for production runs, an efficiency of parallelism of about seventy percent. This result has little statistical significance, but is consistent with our original estimate. Other factors which reduce the speed of calculations are discussed in the body of the report. The estimated values of these additional factors are irrelevant for a future computing system with improved hardware and systems software.

The attempt to validate our method of parallelizing Monte Carlo calculations were unfortunately performed on a computer at the time when its hardware and software were rather unreliable. The tentative conclusion which we can draw from this attempt is not in contradiction with the validity of our approach.

II. EXTRACTING PARALLELISM IN MONTE CARLO

The method we designed is fully described in references 1 and 2. It should be applicable to parallel and vector computers in general, but was particularly designed for radiation penetration calculations on the ILLIAC IV computer.

The ARRAY processor of the ILLIAC IV system consists of sixty-four processing elements (PE's), each with its own memory, all under the control of a single control unit (CU). Data can be transferred between PE memories by an operation called routing. The PE's all carry out the same operation at the same time, as prescribed by the CU, each operating on its own memory. Individual PE's can, however, be disabled for particular operations. With this computer design, and with perfect code organization, if it could be achieved, it would be possible to perform a calculation sixty-four times faster than on a conventional computer with the same hardware component speeds.

The major difficulty with attempting to implement a Monte Carlo code such as SAM-CE (2) on the ILLIAC lies in the intrinsic disorderly nature of Monte Carlo logic. Although considerably modified by importance sampling techniques, a Monte Carlo history is still a computer simulation of the physical events occurring as a particle (neutron or photon) traverses physical media. The order and the nature of these physical events have little, if any, correlation from history to history. The naive approach of following 64 histories simultaneously is therefore not feasible as the parallelism breaks down almost immediately. Our approach is to initiate many histories in each PE, and hold all of them in abeyance until any calculation is required. "Holding the history in abeyance" means that all relevant information about the history - or particle (which we call "latent particle") is to be stored in computer memory. The information consists of data such as position, direction, energy, time, etc., and temporary information, the most important of which is the type of calculation which is to be performed next. Only if enough PE's have at least one latent waiting for a given calculation, will that calculation be performed. The different calculations are performed by completely independent calculational modules. Referring to a conventional flow diagram, each module performs all the calculations involved from start of a branch up to and including the branch point. For example, a module operating on latents just entering collision retrieves and interpolates the absorption cross sections, and samples the absorption probabilities. Another module operates on latents entering scattering and selects the type of scattering to occur, etc. Each module operates on latents of the corresponding type, changes some of the data representing the latents, and changes the state of the latents.

An executive program keeps a tally of latent types, and calls into execution only those modules which have at least one latent in at least N PE's. It is attempted to keep the efficiency criterion N as high as possible, but N is reduced when required.

The parallelism is extracted in problems of any complexity. The cross section data representation can be as exact as wished. Geometrical details can be described using combinatorial geometry[3].

III. VALIDATION OF THE INTERIM VERSION OF THE CODE

The interim version of the code is described in detail in reference 2. Our first task was to make this version operational on the ARRAY processor. This turned out to be the major task of the current project, which exhausted the project funds.

Before describing this effort, it is necessary to give a short description of the operational environment at the Institute for Advanced Computation. It should be noted that our operational experience ended in January, 1976, and does not reflect improvements (if any) after that date.

The operational system includes an impressive configuration of peripheral equipment with the omission of convenient input and output means. The peripheral devices are driven by a powerful control language (ACL). The only means for I/O are either interactive terminals or file transfer via ARPA network. The file transfer works most of the time for card input, but numerous attempts in the course of twelve months were not successful to produce a readable printed output.

The system is peripheral to the main computing device, the ARRAY processor, (AP). The ARRAY processor was down most of the time, and unreliable the remainder of the time. The only I/O capabilities between the AP and the "system" are blocks of 1024 contiguous words in internal format, with no supported software for conversion from internal format to decimal and vice versa.

It should be understood that, given such a system, one can spend quite a bit of time obtaining results from a fully operational code. For instance, a code computing 2x2 sixty four times will wait for execution for a time ranging from a few to many days, and has a very good chance to abort or give wrong answers. We engaged in testing out a code which is somewhat more complex, and which was partially tested out only under translation[2]. The strategy for testing was to move selected intermediate answers into a reserved memory block, and, after storage on the system's disk, examine those data via interactive terminal. After months of perseveration, desperation and determination, a few bugs have been found and corrected. A few of these were traced to the system software, and were kindly corrected by AMES personnel. A few were due to different interpretations of GLYPNIR statements by the GLYPLIT translation and the actual ILLIAC compiler. The remainder were logical errors which did not show up during test runs under translation. Notwithstanding the fact that we strongly suspect that not all branches of the code have been exercised, the testing stopped as soon as a set of reasonable answers was obtained for the single test problem we were nursing. The code was then exercised without modification, and it sometimes reproduced the same answers. The number of Monte Carlo histories was then increased to a few hundred and the code, after a few bad runs, calculated answers which appear to be right - the computed flux being consistent with the analytical flux, within the computed standard deviation. Several attempts to run the code for a few thousand histories were all unsuccessful.

IV. ANALYSIS OF EFFICIENCY

During the design stages of the program, we predicted[1,2] that the ILLIAC version of the code will run twenty times faster than the CDC 6600 computer. We repeat the arguments quoted in Section 9 of reference 1. On the basis of learned estimates, we predicted that useful calculations will be performed with an average PE utilization (e) = 0.7 to 0.8. We recognized the fact that, to achieve that degree of parallelism, there is a heavy overhead associated with routing, searching, etc., with no equivalent on a conventional computer. This overhead reduces the speed of calculations by a factor f which we guesstimated to be of the order of 0.5. Finally, the speed of calculation S_{pe} of a single PE was taken to be equal to the hardware design speed: $S_{pe} \approx S_{66}$, where S_{66} is the actual speed of the CDC 6600 computer.

The overall speed of calculation in the ILLIAC-4 is equal to:

$$S_{14} = e.f.64 \left(\frac{S_{pe}}{S_{66}}\right) S_{66} \qquad (1)$$

Substituting our estimates of e, f, and S_{pe}/S_{66}

$$S_{14} \approx 20\ S_{66} \qquad (2)$$

Our (limited) experience with the "running" version of the code indicates a speed drastically smaller than that given by Equation (2). Although this result indicates that the current version of the code, run with the current state of hardware and software is uneconomical, it is important to attempt to analyze the different factors of Equation (1), as these may be relevant for the design of a variety of Monte Carlo codes for a variety of parallel or pipeline computers, including even an upgraded ILLIAC.

1. The P-E Utilization Factor e

This is the most interesting factor from the point of view of code design. The estimate of the factor has been arrived at in the following manner.

As described in reference 2, the original code has a "debug" feature. Before calling into execution any operational or main overhead module, the executive program examines a logical variable DEB. If true, it first calls a subroutine PR1 and then the module. The subroutine was originally designed to print (in the translated version) the relevant information on module input. The routine has been completely modified. Instead of "printing" anything, it now tallies the number of calls and the PE utilization. The tallies can be examined at the end of the run, and statistical information on the efficiency can be extracted.

As explained in reference 2, the code has been designed to keep the efficiency up while many histories are processed in parallel. The efficiency is expected to deteriorate towards the end of the run when the processing of the bulk of histories has been completed, and only a few histories remain to be processed to their random end. This deterioration is expected to have a negligible effect on the overall efficiency for normal production runs which, as a rule, involve at least several thousand histories. As mentioned before, we were not able to complete any such long runs. For runs consisting of only a few hundred histories, we observed a rather low overall efficiency, and attributed this fact to the "dying stages" of the runs. In order to learn anything about efficiency for production runs, we simply terminated the tally during the dying stages of the run. The results we quote below are based on these truncated tallies.

A grand average over all tallies for all operational modules indicates an overall efficiency of parallelism of 0.72, which is consistent with our original estimate - a rewarding result.

The efficiency of parallelism for individual modules ranges from 0.9 to 0.1, the lower efficiency corresponding to modules less frequently called into execution. This property was built into the design of the code.

As discussed in reference 2, the algorithms utilized by the executive program to keep the efficiency up, can be improved. No major effort in that direction can be justified if the efficiency is indeed 70%. We repeat, however, that this result has been shown only for a single test problem. The test problem is such that minimal amounts of memory are necessary for input data (geometry and cross sections). The room available for latents and minilatents is therefore maximal (25 of each in every P.E.). If that room is reduced, the efficiency is expected to decrease, and some optimization of the executive may become desirable.

2. Processing Element Speed

Being aware of the fact that the ILLIAC-IV was operating at reduced speed and with instruction overlap suppressed, we decided to undertake a fair evaluation of the factor S_{pe}/S_{66}. For that purpose, we made up a problem where both factors e and f are exactly unity, and where routing is not invoked. The problem is to evaluate the sum of 1000 random numbers for 64 different sequences. As the generation of a random number is achieved by multiplication, it is completely parallelized (see Appendix A of reference 2). Comparing running times on the ILLIAC-IV and on a CDC 6600 computer, we obtain $S_{14}/S_{66} = 6.02$. Substituting that result, together with $e = f = 1$ into Equation (1) we obtain

$$S_{pe}/S_{66} = 0.094$$

instead of unity, as we assumed.

3. Overhead Factor

In its current status the interim code includes a number of consistency checks, tallies, interrupts, which are necessary in the current operational environment. This contributes to the overhead and therefore decreases the f-factor, which is currently estimated to be of the order of 0.1. To determine the different components of the overhead, one would have to time separately operational and overhead modules in the course of Monte Carlo execution. This would mean building in interrupts, which would themselves affect the execution time. A less ambitious analysis has been performed. The tallies referred to in subsection 1 above include the number of calls to the main overhead routines. The computer time spent by these routines depends very much on the actual status of latents and minilatents. We "guessed" a typical status, and timed the execution of these routines outside the Monte Carlo code. The only reliable result we obtained from this study is that minilatent routing is the chief contributor to the overhead.

V. CODE IMPROVEMENTS

The interim version of the code[2] included a complete treatment of radiation penetration by the Monte Carlo method. It was designed[1] to treat both neutron and gamma radiation, but was implemented[2] only for gamma rays. The version included no input/output capabilities, the "input" for a single test problem being coded-in as replacement statements. Considerable effort has been applied to complete the implementation in all these areas. The effort has been dropped before completion. A minor effort to improve the overall efficiency of the interim version was partially successful.

1. Improvements in Parallelism

As described in reference 2, the executive program implements given algorithms for maintaining a high degree of efficiency. The algorithms differ in degree of sophistication depending on the type of operational module involved. In Section 8.3 of reference 2, it is recommended that all algorithms be changed to conform with the best one. This has been successfully done in the ILLIAC version. The results of our efficiency study (Section IV) were obtained with the improved version.

2. Reduction in Overhead

As discussed in Section IV.3, it is felt that the major contribution to the high overhead in the operation of the code can be ascribed to the routing of minilatents.

Subroutine MINIRT (described in Section 7.2 of reference 2) has been completely redesigned for greater efficiency. Attempts to make the new version operational were, however, unsuccessful.

3. Incorporation of Input/Output Capabilities

The bulk of the input consists of geometry data and of cross section data. Both sets of data are (successfully) preprocessed in a conventional computer. The data are organized according to the designed[2] layout of the ARRAY memory, and transferred to AMES via ARPA network. Before loading on the ARRAY processor, these files need to be translated to ILLIAC binary format. None of the numerous attempts to use (unreleased and unsupported) system software routines to perform that translation were completely successful.

The bulk of the output consists of fluxes averaged over phase space regions. This output is transferred onto the system's mass storage. It can be successfully examined interactively using an (unreleased and unsupported) system software routine. Several steps are needed to obtain a printed formatted output. The first step is to translate the files from ILLIAC binary to another representation (either decimal or host-binary). Numerous attempts to use (unreleased and unsupported) system software routines were unsuccessful. The next steps were not even attempted: file transfer to a host computer, and formatted output by a (trivial) code (not yet developed) operating on that host computer.

Transfer of ASCII files from AMES to host were achieved with partial success but with enough difficulties to warrant an indefinite postponement of this effort.

4. Inclusion of Neutronics

It has been anticipated that this task would be the major one of the project. Work on this task started in parallel with the installation of the code. The task had, however, been terminated before completion in order to allow work to proceed on the validation of the interim version.

In reference 1, we proposed a complete design of the treatment of neutron interactions. The design was aimed at optimizing the efficiency of the data retrieval modules. The design required a specific memory layout of cross section data - it is given in tables 3, 4 and 5 of reference 1. Some waste of memory can be observed in the storage of angular distributions (table 4) and energy distributions (table 5). Our experience[2] with the detailed design of the code pointed out the importance of reducing the memory requirements. The memory layout has therefore been completely rearranged in the case of angular and energy distributions (at the burden of increasing running time in data retrieval).

A Fortran program has been written to produce such a layout. The input is a "universal" cross section library. The program has been successfully tested out in several (but not all possible) situations.

The ILLIAC retrieval routines have been all coded up and underwent some testing. The testing has not been completed. The subroutines have not been incorporated into the Monte Carlo code.

VI. CONCLUSIONS

The attempt to validate our method of parallelizing Monte Carlo calculations were unfortunately performed on a computer at the time when its hardware and software were rather unreliable. The tentative conclusions which we can draw from this attempt are not in contradiction with the validity of our approach.

VII. REFERENCES

1. H. Steinberg, et al., Design of a Monte Carlo Radiation Transport Code for the ILLIAC IV, DNA 2972F (September 1972).

2. E. S. Troubetzkoy, et al., SAM-IV: A Three Dimensional Monte Carlo Radiation Penetration Code for the ILLIAC IV, DNA 3303F (December 1973).

3. M. O. Cohen, et al., "SAM-CE: A Three Dimensional Monte Carlo Code for the Solution of the Forward Neutron and Forward and Adjoint Gamma Ray Transport Equation", DNA 2830F (July 1972).

DISTRIBUTION LIST

DEPARTMENT OF DEFENSE

Director
Defense Advanced R[illegible]h. Proj. Agency
ATTN: Tact[illegible]hnology Office
ATTN: TIO, [illegible]her
ATTN: Te[illegible]

Defense Documenta[illegible]
Cameron Station
12 cy ATTN: TC

Director
Defense Nuclear Agency
ATTN: RATN
3 cy ATTN: TITL, Tech. Library
ATTN: TISI, Archives
ATTN: DDST

Dir. of Defense Rsch. & Engineering
Department of Defense
ATTN: S&SS (OS)

Commander
Field Command
Defense Nuclear Agency
ATTN: FCPR

Chief
Livermore Division, Field Command, DNA
Lawrence Livermore Laboratory
ATTN: FCPRL

Chairman
Office of Joint Chiefs of Staff
ATTN: SAGA
ATTN: Technical Library

DEPARTMENT OF THE ARMY

Commander
Harry Diamond Laboratories
ATTN: Chief, Nuc. Vulnerability Branch
ATTN: DRXDO-EM
ATTN: DRXDO-TI, Tech. Lib.
ATTN: DRXDO-NP
ATTN: DRXDO-RBF
ATTN: DRXDO-RBG

Commander
Picatinny Arsenal
ATTN: R. Kesselman
ATTN: T. Derosa
ATTN: SARPA-ND-C, P. Angelotti
ATTN: ND-C-T

Director
TRASANA
ATTN: R. E. DeKinder, Jr.

Director
US Army Ballistic Research Labs.
ATTN: Tech. Lib., Edward Baicy
ATTN: DRXBR-X, Julius J. Meszaros
ATTN: DRXBR-AM, W. R. Vanantwerp
ATTN: DRXBR, J. Saccenti

DEPARTMENT OF THE ARMY (Continued)

Commander
US Army Nuclear Agency
2 cy ATTN: Colonel R. Lounsbury

DEPARTMENT OF THE NAVY

Director
Naval Research Laboratory
ATTN: Code 2600, Tech. Lib.

Officer-in-Charge
Naval Surface Weapons Center
ATTN: Code WA501, Navy Nuc. Prgms. Off.
ATTN: Code WA50, John H. Malloy
ATTN: N. E. Scofield

DEPARTMENT OF THE AIR FORCE

AF Weapons Laboratory, AFSC
ATTN: SAS
ATTN: NT, Carl E. Baum
ATTN: DYT, Capt Ray Shulstad
ATTN: SAW
ATTN: SUL

Commander
Tactical Air Command
ATTN: XPS, Capt Powell

USAF School of Aerospace Med., AFSC
ATTN: RA, Chief Radiobiology Div.

ENERGY RESEARCH & DEVELOPMENT ADMINISTRATION

University of California
Lawrence Livermore Laboratory
ATTN: William J. Hogan, L-531
ATTN: Robert Howerton, L-71
ATTN: Tech. Info. Dept. L-3
ATTN: George Staehle, L-24
ATTN: M. Gustavson, L-21
ATTN: R. Barker, L-96
ATTN: Auston Odell, L-531

Los Alamos Scientific Laboratory
ATTN: Doc. Con. for F. P. Young
ATTN: Doc. Con. for E. Chapin
ATTN: Doc. Con. for Donald Harris
ATTN: Doc. Con. for W. Lyons
ATTN: Doc. Con. for R. Sandoval
ATTN: Doc. Con. for T. Dowler

Sandia Laboratories
Livermore Laboratory
ATTN: Doc. Con. for Tech. Library

Sandia Laboratories
ATTN: Doc. Con. for 3141, Sandia Rpt. Coll.
ATTN: Doc. Con. for M. Bunting

US Energy Rsch. & Dev. Admin.
Brookhaven National Lab.
ATTN: Supervisor, Rm. 13 for Nat. Neu. Cross Sec. Sol Pearlstein

ENERGY RESEARCH & DEVELOPMENT ADMINISTRATION (Continued)

US Energy Rsch. & Dev. Admin.
Controlled Thermonuclear Rsch. Div.
ATTN: Doc. Con. for Lester Price

US Energy Rsch. & Dev. Admin.
Division of Reactor Rsch. & Dev.
ATTN: Doc. Con. for Phil Hemmig

Union Carbide Corporation
Holifield National Laboratory
ATTN: Doc. Con. for C. E. Clifford
ATTN: Doc. Con. for Tech. Lib.
ATTN: Doc. Con. for Fred Mynatt
ATTN: Doc. Con. for Rad. Shielding Center

OTHER GOVERNMENT AGENCIES

Central Intelligence Agency
ATTN: RD/SI Rm. 5G48, Hq. Bldg. for
B. Sheffneer, 2922

Department of Commerce
National Bureau of Standards
Center for Radiation Research
ATTN: J. Hubell

DEPARTMENT OF DEFENSE CONTRACTORS

The BDM Corporation
ATTN: Joseph V. Braddock

General Electric Company
TEMPO-Center for Advanced Studies
ATTN: DASIAC

Lovelace Foundation for Medical Education & Rsch.
ATTN: Clayton S. White

Mathematical Applications Group, Inc. (NY)
ATTN: Martin O. Cohen
ATTN: Malvin H. Kalos
ATTN: Herb Steinberg
ATTN: Eugene S. Troubetzkoy

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

McMillan Science Associates, Inc.
ATTN: Robert Oliver

Mission Research Corporation
ATTN: Dave Sowle
ATTN: Conrad L. Longmire

Pacific-Sierra Research Corp.
ATTN: Gary Lang

R & D Associates
ATTN: Cyrus P. Knowles
ATTN: Harold L. Brode

Radiation Research Associates, Inc.
ATTN: Library

Science Applications, Inc.
Chicago Office
ATTN: Dean Kaul

Science Applications, Inc.
ATTN: E. A. Straker
ATTN: W. W. Woolson
ATTN: Marvin Drake

Stanford Research Institute
ATTN: Philip J. Dolan

Vector Research Incorporated
ATTN: Seth Bonder

University of Wisconsin
Nuclear Engineering Department
ATTN: Professor Charles W. Maynard for
Charles Maynard